LOCKHEE

NASA CR-73434-I

REPRODUCIBLE COPY

Study of Systems and Cost/Performance Methodologies for Optimization of Vehicle Assignment

FINAL REPORT

Volume 1

Technical Description

SPERRY RAND

CAT. NO.

50429R-[illegible]3

[illegible]

NASA CR-73434-I

STUDY OF SYSTEMS AND COST/PERFORMANCE METHODOLOGIES FOR OPTIMIZATION OF VEHICLE ASSIGNMENT

FINAL REPORT

VOLUME 1

TECHNICAL DESCRIPTION

8 MAY 1970

PREPARED UNDER CONTRACT NAS2-5202

FOR

MISSION ANALYSIS DIVISION
OFFICE OF ADVANCED RESEARCH AND TECHNOLOGY
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
AMES RESEARCH CENTER
MOFFETT FIELD, CALIFORNIA

BY

LOCKHEED MISSILES & SPACE COMPANY
SUNNYVALE, CALIFORNIA

NASA CR-73434

STUDY OF SYSTEMS AND COST/PERFORMANCE METHODOLOGIES FOR OPTIMIZATION OF VEHICLE ASSIGNMENT

FINAL REPORT
VOLUME 1
TECHNICAL DESCRIPTION

8 MAY 1970

Prepared Under Contract NAS2-5202

By
LOCKHEED MISSILES & SPACE COMPANY
SUNNYVALE, CALIFORNIA

For
MISSION ANALYSIS DIVISION
OFFICE OF ADVANCED RESEARCH AND TECHNOLOGY
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
AMES RESEARCH CENTER
MOFFETT FIELD, CALIFORNIA

[illegible]

FOREWORD

This report volume presents a brief technical description of the analytical approach and solution methodology of a study to develop and implement techniques for optimal assignment of launch vehicles to advanced space missions. This study is being performed for the National Aeronautics and Space Administration under Contract NAS2-5202, and is monitored by Mr. Robert Slye and Mr. Harold Hornby of the Mission Analysis Division of the Office of Advanced Research and Technology.

Individuals of Lockheed Missiles & Space Company, Sunnyvale, California, who contributed to this study are L. F. Fox, project leader; C. J. Golden, key technical member; and M. A. Brunet.

[illegible]

ABSTRACT

During this study, analysis is performed and computer programs developed for optimal assignment of expendable and reusable launch vehicles in a multi-year, multi-mission space program. A primary objective is provision of an analytical tool to help the advance planner to maximize, under restrictive budget constraints, the accomplishment of a future space program. Such a program will normally include existing vehicles, growth versions, and new starts. Explicit handling of all contributing and interrelated costs (recurring, nonrecurring, and sustaining) ensures global solution for a previously intractable problem. The programs analyze the multiple factors that apply, including stage and vehicle characteristics, mission requirements, production aspects, launch sites and pads, sustaining effort, and others. A single computer run identifies stages and vehicles that comprise the optimum launch vehicle family and provides an array of data pertinent to this optimum. An integrated vehicle assignment and budget smoothing feature provides for optimal accomplishment of a space program without exceeding yearly budget constraints.

PRECEDING PAGE BLANK NOT FILMED.

CONTENTS

✓

[illegible]

[illegible]

ILLUSTRATIONS

TABLES

SUMMARY

This document is Volume 1 of a three-volume final report summarizing the results of the Study of Systems and Cost/Performance Methodologies for Optimization of Vehicle Assignment. Volumes 2 and 3 provide details on developed computer programs. This volume contains a brief technical description of the analytical approach and solution methodology.

During this study, an analysis is performed and computer programs are developed for optimal assignment of expendable and reusable launch vehicles to payloads in a multi-year, multi-mission space program. The primary objective is to provide an analytical tool to help the advance planner maximize, under restrictive budget constraints, the accomplishment of a total space program over an extended future period.

An important result is the solution of a previously intractable problem – namely, to provide an optimal assignment of vehicles to missions for the least total program cost. The developed program explicitly handles all contributing costs, including recurring, nonrecurring, and sustaining types that are associated with stages and vehicles, and their integration, alternate launch site and pad facilities and other parameters. These cost factors are interrelated. Thus, changes in one vehicle-to-mission assignment influences all other assignments, resulting in a large combinatorial problem. Explicit handling of these cost categories ensures a global optimum solution.

In operation, the complete program (1) analyzes stage characteristics and integrates stages into feasible vehicles (user option). (2) screens vehicle performance against mission requirements, (3) incorporates learning curve effects and batching in manufacture (4) provides for the use of alternate launch sites and pads, and (5) evaluates reusable vehicle sizes and incorporates other related parameters. The

output solution identifies the optimum vehicle-to-mission assignment and provides an array of associated data. For sensitivity analyses of the various parameters that enter the problem, in addition to the optimum solution, the program can identify the N next best solutions ordered by total cost. By simultaneous evaluation of alternately configured stages, the program can be used for comparative design analyses.

The program is integrated with a budget-smoothing program. Therefore, the optimum vehicle-to-mission assignment can be adjusted so that the total space program can be accomplished without exceeding planned yearly budget constraints. The program also is structured to provide flexibility to the user in selecting only subroutines that are needed for his particular problem.

Section 1
INTRODUCTION

1.1 BASIC PROBLEM

In seeking cost reductions, the mission planner is faced with the inherent complexity of space programs. It is frequently impractical to manually assess all of the interrelated factors in the accomplishment of a multi-year, multi-mission space program and concurrently ensure that all applicable constraints are satisfied.

One such previously intractable problem is that of determining the optimal assignment of launch vehicles to space missions for the least total program cost. Factors considered in this assignment include stage and vehicle physical and performance parameters, and costs in three basic categories — namely, recurring, nonrecurring, and sustaining. These costs are directly related to the accomplishment of each mission in the mission model. If only recurring costs were involved, the problem could be solved simply. However, when non-recurring and sustaining costs are included, any change in the vehicle-to-mission assignment changes the total program cost because of the interrelationship between these costs.

In a typical example, the consideration of all possible vehicle-to-mission assignments for a nominal space program having 20 possible launch vehicles and 300 missions over a 15-year period would require the analysis of combinations on the order of 20^{300}. Thus, a basic problem encountered in constructing a launch vehicle assignment model is combinatorial in that all possible assignments cannot be evaluated for a problem of realistic size.

1.2 SOLUTION APPROACH

Different analytical approaches may be employed to avoid this combinatorial problem. One of the most promising approaches is to apply a modified form of the branch-and-bound technique developed for the related "Traveling Salesman Problem," by Little, et al. (Ref. 1). References 2 and 3 also provide related background information.

Little's basic algorithm must first be modified to handle assignment type problems with realistic cost functions, since fixed, non-recurring assignment costs exist. These latter costs are only incurred the first time a vehicle is selected to perform a mission. A further algorithm modification is necessary since vehicle-related costs are intradependent (e.g., the "Atlas family," "Titan family," the "Agena family" share certain costs), so there is not a simple one-to-one relationship between development costs and launch vehicle candidates.

The most important and extensive modifications that must be made, however, are those which permit the explicit consideration of the time-dependent costs ("annual operating" cost) associated with a launch vehicle.* For realistic launch vehicle use lifetimes, these time-dependent costs are typically of the same magnitude as non-recurring costs; therefore, they cannot be handled with accuracy by iterative techniques. In fact, cases in which iterative methods converge to demonstrably erroneous solutions have been found. Time-dependent costs are sufficiently large in real space program examples that iterative techniques guarantee only a relative minimum, not an absolute minimum cost assignment.

1.3 GENERAL MODEL LOGIC

The algorithm developed during this study and described herein handles all applicable costs explicitly and therefore an absolute minimum cost assignment is guaranteed. Figure 1-1 illustrates the general flow diagram for the assignment model where the

*For example, sustaining engineering costs, annual launch service costs, guidance support costs, and launch complex operation or maintenance costs.

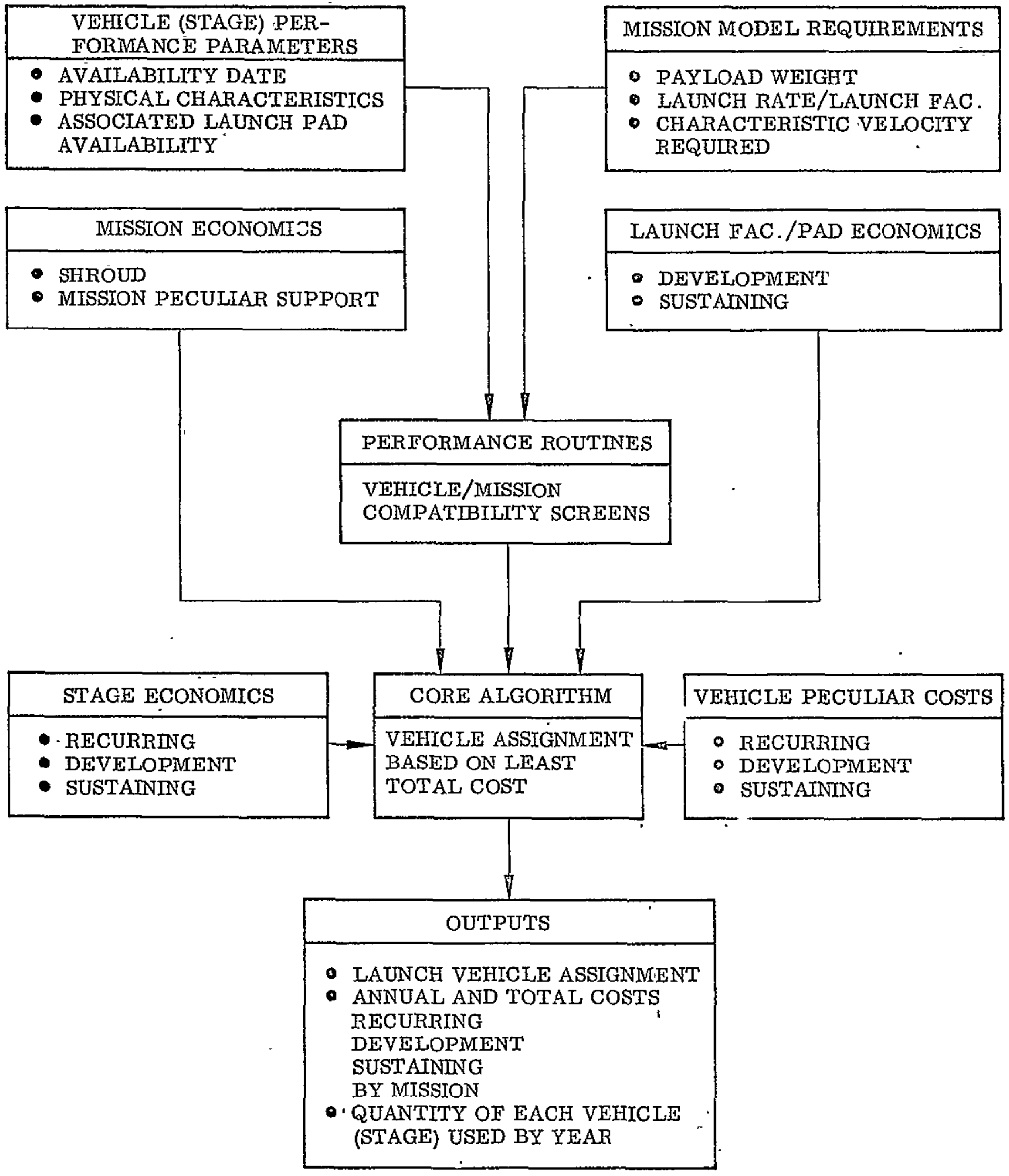

VEHICLE (STAGE) PERFORMANCE PARAMETERS
• AVAILABILITY DATE
• PHYSICAL CHARACTERISTICS
• ASSOCIATED LAUNCH PAD AVAILABILITY
MISSION MODEL REQUIREMENTS
• PAYLOAD WEIGHT
• LAUNCH RATE/LAUNCH FAC.
• CHARACTERISTIC VELOCITY REQUIRED
MISSION ECONOMICS
• SHROUD
• MISSION PECULIAR SUPPORT
LAUNCH FAC./PAD ECONOMICS
• DEVELOPMENT
• SUSTAINING
PERFORMANCE ROUTINES
VEHICLE/MISSION COMPATIBILITY SCREENS
STAGE ECONOMICS
• RECURRING
• DEVELOPMENT
• SUSTAINING
CORE ALGORITHM
VEHICLE ASSIGNMENT BASED ON LEAST TOTAL COST
VEHICLE PECULIAR COSTS
• RECURRING
• DEVELOPMENT
• SUSTAINING
OUTPUTS
• LAUNCH VEHICLE ASSIGNMENT
• ANNUAL AND TOTAL COSTS
RECURRING
DEVELOPMENT
SUSTAINING
BY MISSION
• QUANTITY OF EACH VEHICLE (STAGE) USED BY YEAR

Fig. 1-1 General Model Flow Diagram

cost minimizing algorithm forms the heart of the model. Mission model requirements include payload weight and characteristic velocity requirements, various stage and vehicle performance requirements, proposed launch rate and site, and other related characteristics. Vehicle performance capabilities are specified directly and are compared to mission requirements to determine which vehicles can and which cannot perform a given mission. Availability of launch vehicles and appropriate launch facilities are among the parameters considered in this screen. This capability data, along with economic data, is input to the optimization algorithm which determines the least cost assignment for the total program. Data is input in terms of stages wherever possible, since this practice allows the various types of cost sharing interactions to be considered. The output includes various breakdowns in the optimum vehicle-mission assignment and can identify the next N best solutions for sensitivity analysis. For simplicity only typical data are indicated on Fig. 1-1. Complete inputs and outputs to the algorithm are discussed in detail in Section 3 and Appendices A and B.

This computer program has been integrated with a previously developed budget smoothing model (Ref. 4) so that the combined program outputs a least cost total space program constrained by realistic funding levels. Section 4 and Volume 3 describe this integrated program in detail.

Section 2
MULTI-BRANCH ALGORITHM

As indicated in Section 1, a significantly modified version of a branch and bound algorithm which provides for multiple branching at nodes is a key analytic procedure in the program. This control algorithm is described in some detail in this section.

2.1 ORGANIZATION OF INPUT

Consider m missions in a mission model to be performed by at most n candidate launch vehicles. Each vehicle is specified by its stage components each of which may belong to one of several families. Table 2-1 describes the type of input cost data which is required. A cost must belong to one of the three categories, but may be related to any of the sources listed or to a particular vehicle. A family is essentially a shared cost group of stages which share either components, launch facilities or maintenance. Interstage integration may be between individual stages or between families of stages or any combination of these two.

Each mission is specified by a list of requirements, including launch site restrictions, payload weight and diameter, characteristic velocity, and annual launch rate, along with payload costs and mission-peculiar costs. The recurring cost, c(i, j), is based on complete vehicle and mission information. It is defined as the expected operational cost of performing the j-th mission with the ith vehicle so that

$$c(i, j) = X(j)\, L(j)\, [P(j) + R(i)] \qquad (2.1)$$

where

$X(j)$ = priority of mission j $0 \leq X \leq 1$

$L(j)$ = desired number of launches for j-th mission

$P(j)$ = payload cost of mission j

$R(j)$ = recurring cost of vehicle i = sum of component recurring costs associated with vehicle i

Table 2-1

DESCRIPTION OF INPUT COST DATA

Cost Related to
● Stage
● Family of Stages
● Interstage Integration
● Launch Facility
Categorized in Terms of
● Unit (Recurring)
— Stage Procurement
— Launch Propellants and Services
— Interstage Adapter
— Guidance
● Fixed (One-Time)
— Basic Development or Uprating Cost
— Pad Conversion
— Subsystem Integration
— Launch Complex Facility and GSE Procurement/Modifications
● Annual
— Sustaining Engineering
— Guidance Support
— Complex Operation and Maintenance
— GSE Maintenance
— Government Administration Service

To make the model realistic and to eliminate computations on vehicle-mission combinations which are not usable, vehicle-mission assignments are first computer screened for feasibility. c(i, j) is assigned a large positive number if, for any reason, the i-th vehicle cannot perform the j-th mission.

The list of all annual and development cost items to be considered form a matrix of budget options (See Tables 2-2 and 2-3), which are "tied" to candidate vehicles so that all budget options associated with a particular vehicle must be included in order for that vehicle to be eligible for assignment to any mission.* For example, if Vehicle 1 consisted of Stages 1 and 3 and Vehicle 2 consisted of Stages 1, 3 and 4, then elimination of Stage 4 development (Budget Option 3) would eliminate Vehicle 2 from consideration but not Vehicle 1. Elimination of integration costs between Stages 1 and 3 (Budget Option 6) would eliminate both Vehicles 1 and 2 from future consideration.

This procedure of using budget options upon which to base decisions and then relating these budget options back to vehicle availability for selection exploits the unique aspects of the problem. Instead of setting up the vehicle assignment problem to fit the original algorithm, the powerful features of the branch-and-bound method were adapted to fit the realistic situation. Thus, the assignment resulting from this model achieves global minimum cost while considering the various economic interactions between vehicles.

2.2 SOLUTION TECHNIQUE

This technique in general employs a logical search of the space of all feasible solutions. The solution space is repeatedly partitioned into smaller and smaller subsets, and a lower bound is calculated for the cost of all solutions

*Budget options may include stage development and annual costs, stage integration costs, "family" type costs and launch pad modification costs.

Table 2-2

LIST OF TYPICAL BUDGET OPTIONS

Budget Option Number	Type	Fixed (Development)	Annual (Sustaining)
1	Stage 1	15	2
2	Stage 3	100	30
3	Stage 4	2	1
4	Shared Cost 1	10	—
5	Shared Cost 2	0	1
6	Integration Between Stages 1 and 3	15	5

Table 2-3

TYPICAL VEHICLE-BUDGET OPTION RELATIONSHIP

Vehicle Number	Stage Component Numbers	Applicable Budget Option Numbers
1	(1, 3)	1, 2, 5, 6
2	(1, 3, 4)	1, 2, 3, 4, 5, 6
3	(2, 3)	2, 4, 5

1-4

within each subset. After each partitioning, those subsets with a bound that exceeds the cost of a known feasible solution are excluded from all further partitionings. The process continues until a feasible solution is found (no more partitioning is possible in this set) such that its cost is no greater than the bound for any other subset.

For the vehicle to mission assignment problem, this process is most easily visualized with a tree as shown in Fig. 2-1. For simplicity in general process description, nomenclature in Fig. 2-1 is independent of that in Tables 2-2 and 2-3.

At the top of the tree all vehicles are available for assignment so a lower bound is calculated based upon the vehicle to mission assignment which minimizes total vehicle recurring cost, and at the same time satisfies launch vehicle availability and mission requirements. All non-recurring costs are ignored at this first node. If annual costs are zero for budget option A, (e.g., Stage A has development cost only) then two branches are generated from the first node; one branch represents all solutions which require option A to be selected, $\textcircled{A}$, and the other represents all solutions where option A is not necessary, $\textcircled{\bar{A}}$. Lower bounds are then computed for each of these two nodes. For the branch assuming selection of option A, the recurring cost lower bound will remain the same as for the preceding node since no vehicles have been excluded from consideration. However, the development cost associated with selecting option A, (D_A), must be added to this recurring cost bound for a total lower bound estimate. The branch excluding option A has no non-recurring cost associated with it so $D = 0$, but since all vehicles dependent upon that option must now be excluded from consideration in the calculation of the minimum recurring costs this minimum will be a member of a monotonically increasing series, yielding a new lower bound, R_2 , for this node.

The branch with the least lower bound is chosen as the most promising for a new partition. Assume $LB_3 < LB_2$ in Fig. 2-1. If option B has an associated annual cost S_B and the mission model duration is 2 years, then there are 3 branches generated

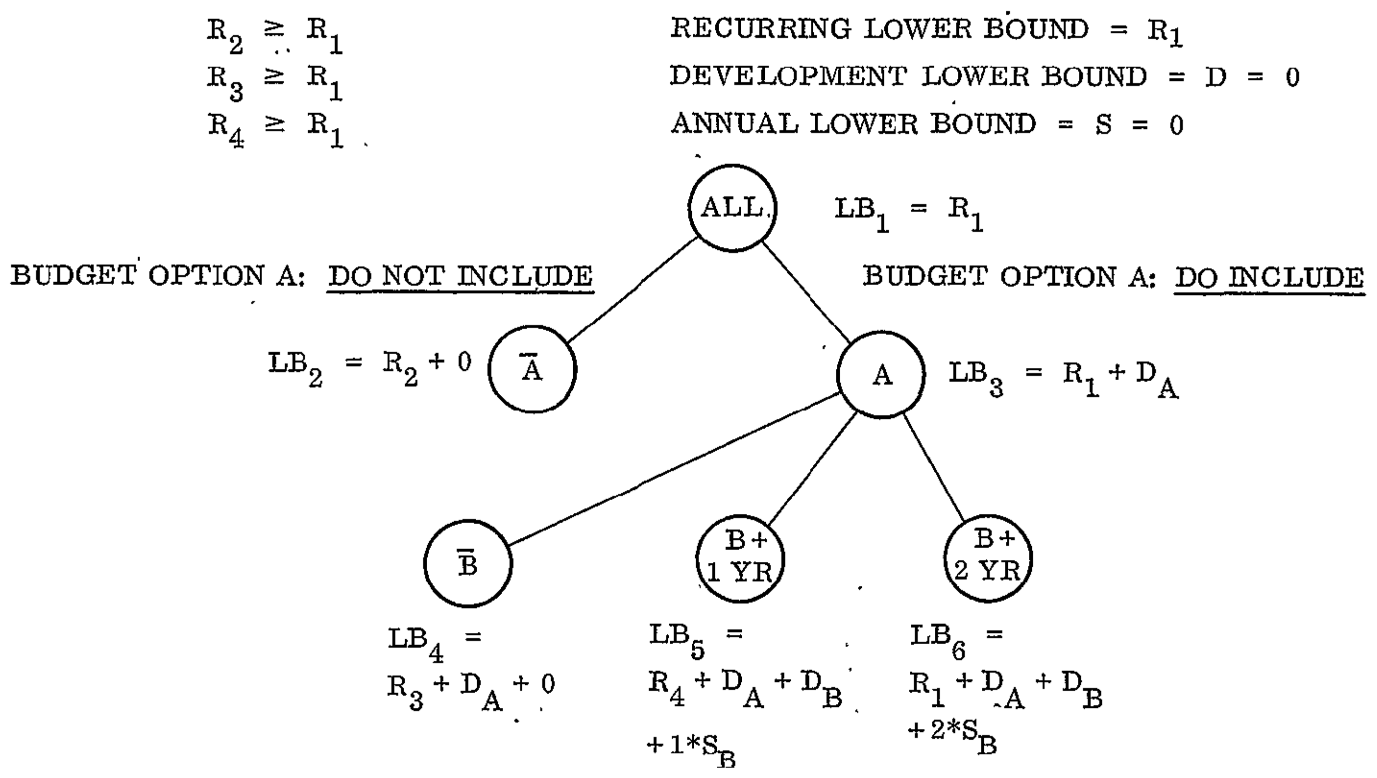

$R_2 \geq R_1$
$R_3 \geq R_1$
$R_4 \geq R_1$
RECURRING LOWER BOUND = R_1
DEVELOPMENT LOWER BOUND = D = 0
ANNUAL LOWER BOUND = S = 0
ALL
$LB_1 = R_1$
BUDGET OPTION A: DO NOT INCLUDE
BUDGET OPTION A: DO INCLUDE
$LB_2 = R_2 + 0$
$\bar{A}$
A
$LB_3 = R_1 + D_A$
$\bar{B}$
B + 1 YR
B + 2 YR
$LB_4 = R_3 + D_A + 0$
$LB_5 = R_4 + D_A + D_B + 1*S_B$
$LB_6 = R_1 + D_A + D_B + 2*S_B$

Fig. 2-1 Decision Tree

from this last node; one branch represents all assignments which include option A but not option B, $\bigcirc\!\!\!\!\!\;\bar{B}$, one branch includes options A and B and sustains B for the first year it is available only, (B + 1 yr), and the last branch also includes options A and B, but sustains B for the total mission model duration, (B + 2 yr). In general there are N + 1 branches generated at each node, where N represents the period in years which the mission model covers. Lower bounds are computed for each of these new branches as before except that the appropriate amount of annual costs for option B must be added to each lower bound. If N > 10, then there are $\left[\frac{N+3}{2}\right]$ branches generated at each node. Each branch represents sustaining the option for an even number of years. This process is continued, always partitioning from the branch with the least lower bound from all branches not yet considered, until a solution is reached, i.e., until all of the budget options have been considered and either excluded or included for a specific period of time. If this solution also has a bound less than or equal to the bounds established for all other uncompleted branches, then this solution is the optimum solution. If there are bounds on other uncompleted branches which are less than the bound on the solution just found, then the partitioning process continues from these uncompleted branches until a solution is determined whose bound is less than or equal to all other calculated bounds.

In general, the number of branches which must be generated depends upon the number of budget options to be considered, the number of years covered by the mission model, and the number of "almost optimal" solutions to the problem. The number of budget options determines the height of the decision tree and the number of "almost optimal" solutions determines the spread of the tree. If the entire tree were generated, all possible combinations of costs and assignments would have to be computed and nothing would be saved by this method over direct enumeration. In practice, only a small portion of the tree is generated since the lower bound at any branch is a lower limit on any final solution generated by branching from that point. Therefore, no branch need be extended past the point at which its lower bound exceeds the value of a known solution.

2.3 INCREASING EFFECTIVENESS

By choosing an effective criterion to determine which budget option should be considered next at any node and by calculating a lower bound at each node which approximates closely the actual lowest cost of all solutions in that set, the size of the decision tree, and hence the calculation time, is reduced considerably. These two techniques will now be discussed for the vehicle assignment program.

2.3.1 Branching at a Node

Associated with each node are three sets of non-recurring cost "budget options:" those that have been deleted, those that have been retained and sustained for a specified number of years, and all remaining "options" whose status has not yet been determined. In order to select which of the undetermined "options" should be considered next, a penalty function is defined for each such option, based upon the magnitude of the components of that option and the increase in recurring cost which would result if that option were not retained. The option which has the largest penalty associated with it is selected for consideration. That is, the option which is most likely to be included in the final optimum selection and which provides the largest difference in lower bound values is chosen (Principle of Inclusion). This choice was determined heuristically; however it results in the lower bound approaching the optimum mission model lower bound quickly. Nodes with very large bounds are also quickly discarded using this method so less computer storage is required. In practice, this principle of inclusion often leads directly to an optimal solution.

2.3.2 Calculation of Lower Bound

A conservative estimate of the total program cost may be made at each node simply by summing the development cost-options which are definitiely included at that node, and their corresponding annual costs multiplied by the number of years which each is to be available. To this number is added the recurring cost bound determined by assigning the least expensive vehicle, not already excluded, to each mission on the basis of recurring cost only. Since each is a lower bound to the true non-recurring or recurring cost, their sum represents a lower bound approximation to the total mission model cost.

$$Z = \text{Lower Bound} = \sum_{j=1}^{m} \min_{i} c(i,j) + \sum_{i=1}^{p} NR(i) \qquad (2.2)$$

where

$NR(i) = D(i) + S(i)(t_f - t_a)_i$ if option i is included at this node and through year t_f. (t_a is the first year option i is available.)

$NR(i) = 0$ if option i is deleted or its status is undetermined.

A lower bound (LB) penalty function may be calculated for each undetermined cost in order to sharpen the lower bound and hence reduce the number of nodes which must be considered. Great care must be exercised in the choice of this function, since the lower bound represents the cost of a corresponding less constrained problem. In order to insure optimality, this estimated cost must be less than the cost of any assignment solution found by continuation from that node.

The LB penalty function which works effectively for the launch vehicle to mission assignment problem is based on the increase in recurring cost which results if each vehicle currently assigned to a mission is deleted. The increased recurring cost for each vehicle is compared to the development cost of that vehicle plus one year annual cost, (since if a vehicle is developed it must be sustained for at least one year). The smaller of these two numbers is taken as the penalty associated with deletion of that vehicle. Only the maximum penalty over all vehicles is added to the conservative lower bound because vehicle development costs are not independent and hence the sum of these penalties added to the conservative lower bound would in general be larger than a true lower bound value. If a true lower bound is not used in the algorithm, a global optimum solution cannot be guaranteed. The use of the above penalty function reduces computer run-time by one-third or more over the conservative case.

Although total calculation time for a given vehicle-to-mission model is directly influenced by the complexity of the penalty function, there is a tradeoff region which produces minimum total computer time for a given case.

Figure 2-2 illustrates a typical tradeoff relationship between total calculation time and penalty complexity. Point B represents direct enumeration of all possible assignments since the branch-and-bound technique requires some criterion upon which to choose the next option for consideration. Point C represents a random selection of the next option for consideration, which is extremely inefficient, but still represents an improvement over direct enumeration. Point D represents calculation of minimal assignments at each node, in order to take full advantage of the lower bound approximation. However, calculation time at each node would be excessive in this case. The area of interest (shaded) represents the combination of an effective criterion for choosing which option should be considered next and a good approximation to the actual lower bound cost at each node.

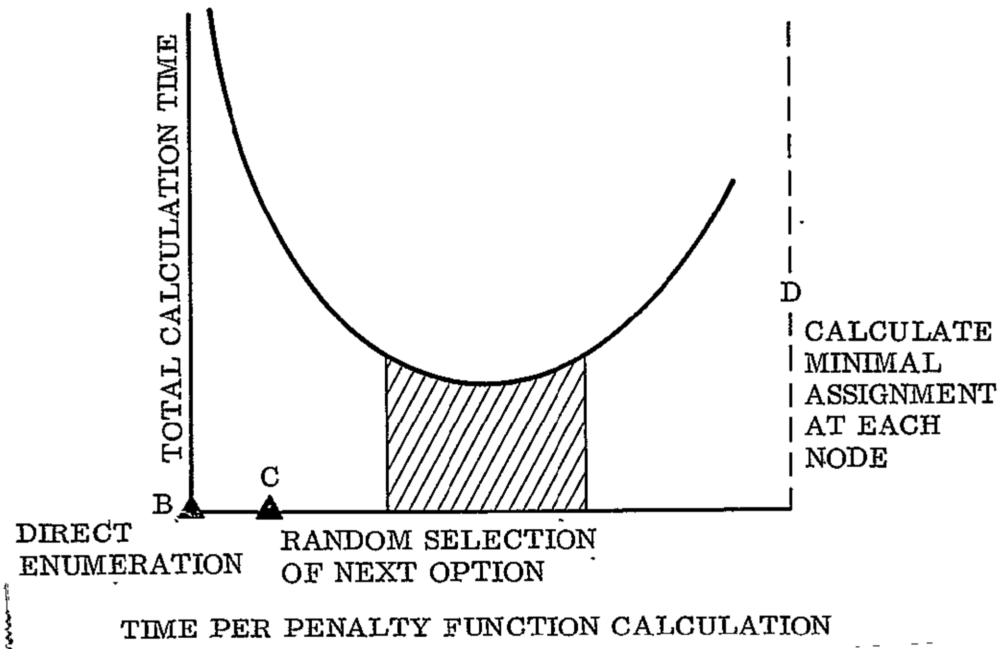


Fig. 2-2 Effect of Penalty Function Complexity on Total Calculation Time

2.4 OPTIMALITY OF SOLUTION

Branching from the initial node continues until all costs associated with a feasible solution have been considered at some node. The total cost of the solution at this node is compared to all lower bounds on other (incomplete) nodes. If some nodes have lower bounds that are exceeded by the cost of the solution just found, then the node with minimum lower bound is chosen as the starting point for reapplication of the branching procedure. This process is continued until there are no nodes whose lower bounds are less than the cost of the optimum solution found so far. For a given set of mission requirements, all possible combinations of vehicle stages into launch vehicles and all possible assignments of these launch vehicles to the given missions are represented by the very large number of terminal nodes of the decision tree. As one proceeds from the initial node to any given terminal node, the sequence of lower bounds obtained is non-decreasing. Therefore, when the final least lower bound condition occurs, the terminal node represents the global minimum cost launch vehicle combination and its assignment.

Section 3
LAUNCH VEHICLE ASSIGNMENT PROGRAM

This section contains a description of the computer program based on the algorithm described in Section 2. The program logic is described in general in this Section 3 and is detailed in Appendix C, Volume 2. Detailed input requirements are listed in Appendix A along with a glossary of input terms. A sample case is presented in Appendix B which illustrates form of output and may be used for program checkout.

3.1 LOGIC

The overall program logic is shown schematically in Fig. 3-1 and consists of 15 subroutines. Each subroutine has been constructed as a self-contained package with a minimum of interrelationship between routines. Consequently, any subroutine can be altered, expanded, or modified with the minimum amount of effort. The length of each subroutine was restricted so that maximum use of the Fortran H mode of compilation would result. This extremely efficient mode of compilation results in reduced storage and reduced run times in comparison with the more common Fortran G mode.

Figure 3-2 illustrates the overall relationship between subroutines. The primary purpose of each subroutine is presented at the beginning of its respective listing in Appendix D. Each subroutine is stored with the prefix MOX02 followed by its two key letters; for example, MAIN is stored under MOX02MN. Subroutine PACK is the only exception to these subroutine storage identifiers. It was written by the Technical Monitor for general NASA use and is stored under the code MOX01PK. As illustrated in Fig. 3-2, the program is currently run with one overlay necessitated by storage limitations.

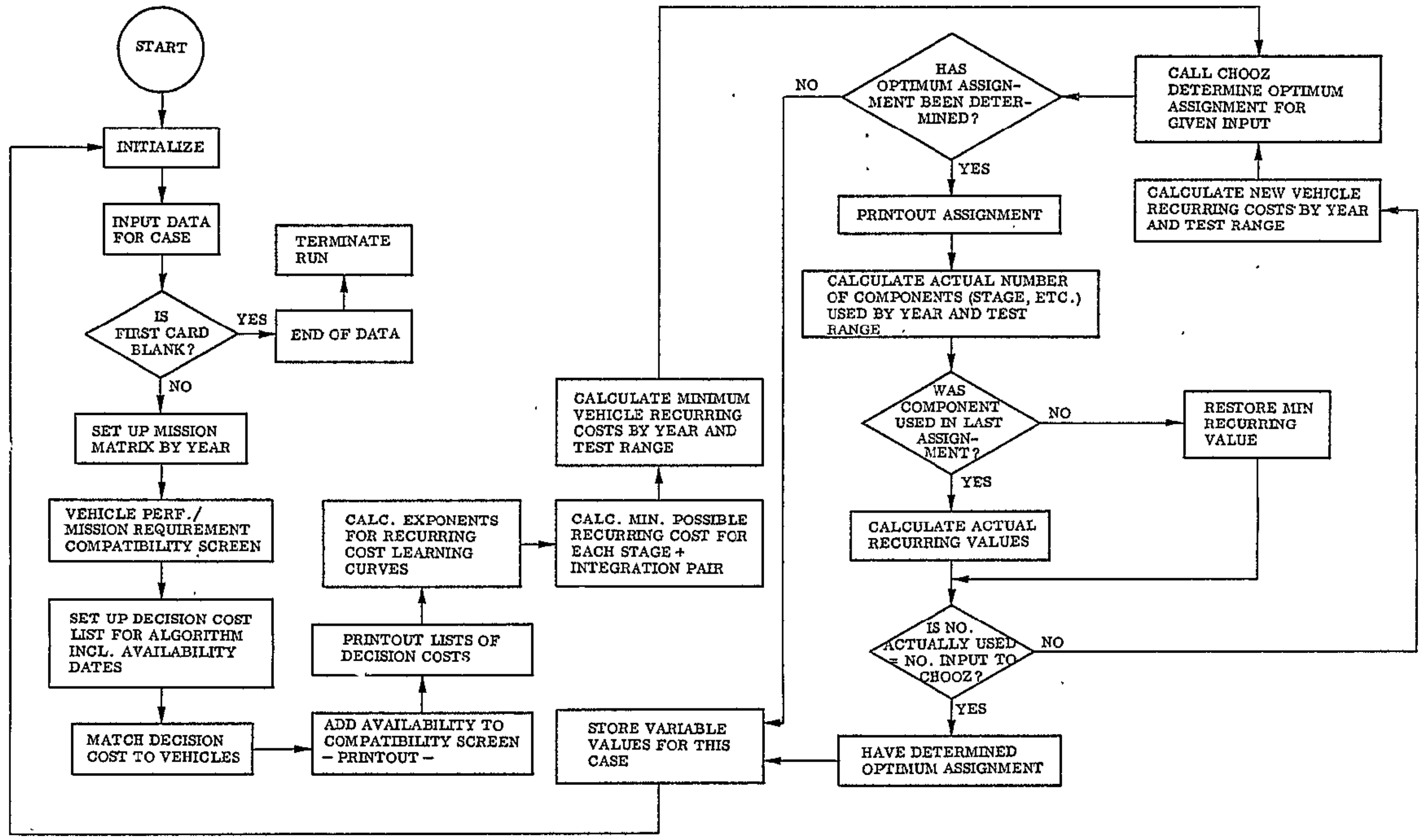

START
INITIALIZE
INPUT DATA FOR CASE
IS FIRST CARD BLANK?
YES
NO
END OF DATA
TERMINATE RUN
SET UP MISSION MATRIX BY YEAR
VEHICLE PERF./ MISSION REQUIREMENT COMPATIBILITY SCREEN
SET UP DECISION COST LIST FOR ALGORITHM INCL. AVAILABILITY DATES
MATCH DECISION COST TO VEHICLES
ADD AVAILABILITY TO COMPATIBILITY SCREEN – PRINTOUT –
PRINTOUT LISTS OF DECISION COSTS
CALC. EXPONENTS FOR RECURRING COST LEARNING CURVES
CALC. MIN. POSSIBLE RECURRING COST FOR EACH STAGE + INTEGRATION PAIR
CALCULATE MINIMUM VEHICLE RECURRING COSTS BY YEAR AND TEST RANGE
STORE VARIABLE VALUES FOR THIS CASE
HAS OPTIMUM ASSIGNMENT BEEN DETERMINED?
NO
YES
CALL CHOOZ DETERMINE OPTIMUM ASSIGNMENT FOR GIVEN INPUT
CALCULATE NEW VEHICLE RECURRING COSTS BY YEAR AND TEST RANGE
PRINTOUT ASSIGNMENT
CALCULATE ACTUAL NUMBER OF COMPONENTS (STAGE, ETC.) USED BY YEAR AND TEST RANGE
WAS COMPONENT USED IN LAST ASSIGNMENT?
NO
YES
RESTORE MIN RECURRING VALUE
CALCULATE ACTUAL RECURRING VALUES
IS NO. ACTUALLY USED = NO. INPUT TO CHOOZ?
NO
YES
HAVE DETERMINED OPTIMUM ASSIGNMENT

Fig. 3-1 General Flow Diagram for Assignment Program

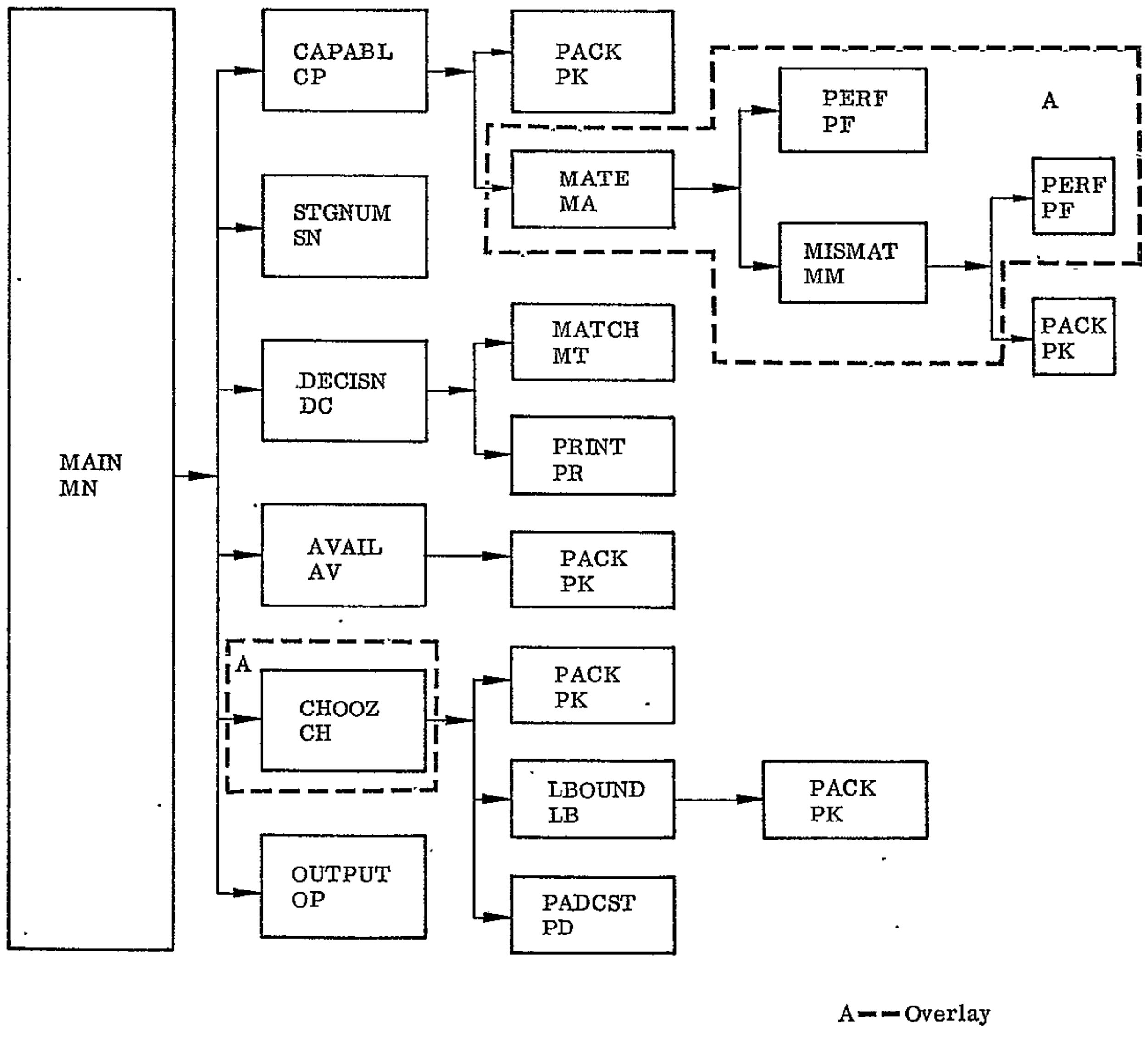


Fig. 3-2 Program Subroutine Relationships

Dimension constraints are detailed for all input variables in Appendix A which also includes comments on internal restrictions involving variables. All other dimension constraints, data statements and equivalence relations may be found at the beginning of the MAIN Fortran listing in Appendix D.

3.2 GENERAL INPUT REQUIREMENTS

Detailed input requirements are included in Appendix A and a general description of cost terms used in the model was presented in Table 2-1 of the preceding section. Input cost data may be related to individual stages, a family of stages, interstage integration ~~on~~ or launch facility. At present a launch facility is either ETR or WTR while a launch complex consists of at most three specific pads, at one of these facilities. Unit or recurring costs are expended at each launch. Fixed or one-time costs may be spread-out over a period of years, but are only spent once and must be spent before the component is considered operational. Annual costs include all sustaining-type costs, and represent any cost computed on a yearly basis. All input costs are grouped into one of these three categories, but may be related to any of the sources listed or to a particular vehicle if more convenient. X

Table 2-1 categorizes typical costs which arise in a vehicle-to-mission assignment and identifies the meaning of the three terms (unit, fixed, and annual). X

1.3.1 The actual relationships used in the program between cost categories and types are shown on Table 3-1. Recurring costs are input in terms of stage or integration costs. Any recurring costs associated with a specific launch pad are input under the associated booster recurring costs. Vehicle recurring costs are then computed by the program as the sum of all component stage recurring costs plus any applicable integration recurring costs.

Table 3-1

INPUT COST CATEGORIES

- Unit (Recurring)
  - Stage — Hardware; Launch Site Support
  - Integration — Hardware
- Annual
  - Shared Cost Group, Stage, Integration — Hardware; Launch Pad; Launch Site
- Fixed (One-Time)
  - Shared Cost Group, Stage — Hardware; Launch Pad
  - Integration — Hardware

Annual costs are quite complex in nature and hence require a detailed format for introduction into the program. For instance, fixed launch pad costs are launch complex oriented while annual costs are stage oriented. Annual costs are further complicated by the fact that a second pad does not require the same number of people to maintain it as the first pad. Discipline personnel are not fully utilized with only one pad and thus need not be duplicated for the second pad. However, other workers cannot maintain two pads at once, so they must be duplicated for the second pad.

Fixed costs may be entered for shared cost groups (families) or individual stages for hardware and/or launch pad expenditures. Integration costs are only hardware oriented for the majority of real cases, so this one category of input is sufficient.

3.3 RATE EFFECTS

The model is capable of investigating the implications that learning curves, batch production, and other types of rate effects have on the optimum launch vehicle assignment.

3-5

Each stage now has three types of recurring costs associated with it, and for flexibility in handling differing costing methods, each of these can be input in one of two forms. The first type includes costs which are not related to launch-site, such as hardware costs. The second type refers to launch-support costs associated with the eastern test range, and the third refers to launch support costs associated with the western test range. Each of these three cost categories can be input either in learning curve form or in jump-discontinuous form.

3.3.1 Learning Curve Form

The cost of the first unit, C_1, (either stage or integration related) is input along with the learning curve percentage, p . The average cost of producing the N^{th} unit, $\bar{C}_N$, is then determined by

$$\bar{C}_N = C_1 N^{-K} \quad , \text{ where } -K = \frac{\ln p}{\ln 2} .$$

f, 3-3 This equation utilizes the log-linear cumulative average form of learning curve. This type of learning curve is illustrated in Fig. 3-3.

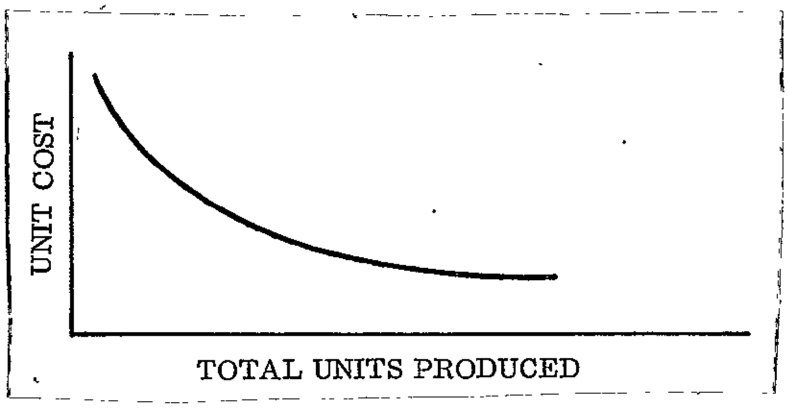


Fig. 3-3 Learning Curve Form

All units required in any given year are assumed to be produced at one time unless batching is specified.

Corresponding program variables are indicated in Table 3-2.

Table 3-2

CORRESPONDING VARIABLE NAMES — LEARNING CURVE FORM

Equation Name	Program Name	Comment
C_1	SR	Stage Related
	RINT	Integration Related
p	PLC	Stage Related
	PLCINT	Integration Related

3.3.2 Jump-Discontinuous Form

This type of input can be used for those stage recurring costs which do not easily fit into the learning curve form. It is assumed that the total cost of x stages, for $1 \leq x \leq P$, is a constant, C. For $x > P$, the total cost, T, is defined by

$$T = mx + b .$$

The input variables are P, C, m, and b. Their corresponding program names used in the stage input data section are indicated on Table 3-3. The average cost of producing the N^{th} unit is determined by dividing either C or T by N, whichever is applicable depending on the total number of units produced. Figure 3-4 shows this form of learning curve.

T 3-3

f 3-4

Table 3-3

CORRESPONDING VARIABLE NAMES –
JUMP – DISCONTINUOUS FORM
(STAGE RELATED ONLY)

Equation Name	Program Name
P	POJ
C	SRJ(1)
m	SRJ(2)
b	SRJ(3)

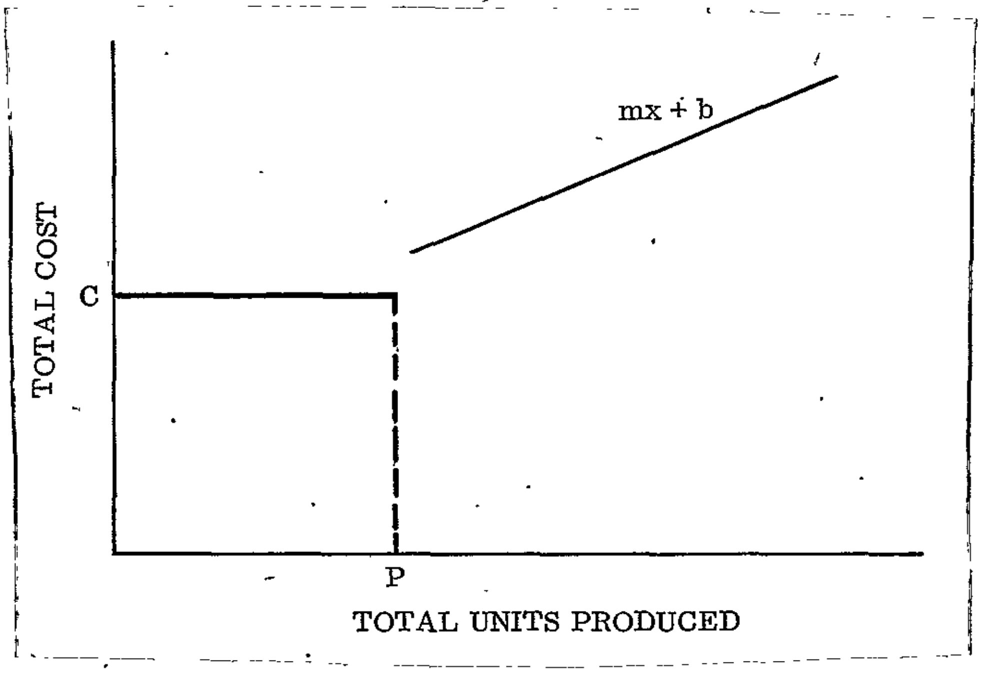


Fig. 3-4 Jump – Discontinuous Form

3.3.3 Batch Production

The effects of batch production are handled by inputting a number with each stage to indicate the number of years over which batching will occur. Then all the units of each stage required for launches during that range of years are assumed to be produced at one time, thus resulting in a lower cost when the learning curves described above are applied.

3.3.4 Setting Up Initial Case

Since the use of learning effects assumes that the number of each unit required has already been determined and since these are the numbers which the model is optimizing, the incorporation of learning effects into the model requires that it now iterate on recurring costs. For the first iteration each stage and integration is given an equal opportunity to be chosen on the basis of recurring costs — i.e., it is assumed that each stage will perform every launch for which it is capable in the model. Hence, the maximum feasible number of each stage and integration as a function of year and test range is used to determine which point on the recurring cost curve to use. The unit recurring cost for each vehicle by year and launch site, which is the sum of the unit recurring costs of its associated stages and integrations, is then the lowest possible for the mission matrix being optimized. The best assignment for the first iteration is then determined based on these costs.

3.3.5 Determination of Optimum Solution

For subsequent iterations the number of each stage and integration used in the last iteration by year and launch site is used to determine the new point on the recurring cost curve, except for those stages and integrations which were not selected for missions. To give these unused stages an opportunity to be selected on the next iteration, the previous minimum value of the recurring cost is reused. New vehicle recurring costs are calculated using these new or restored values and a new best assignment of vehicles to missions is determined. Between iterations a comparison is made of the number of each stage used in the previous iteration to that to be used in the next. If these are equal in every case then the optimum solution is that found in the last iteration. All cases tested using this refinement have converged to the optimum solution in two iterations. This behavior is expected in general.

3.4 LAUNCH VEHICLE TO MISSION COMPATIBILITY SCREEN

Considerable saving of computational space can be realized if the very large set of all possible vehicle-mission combinations is first screened for compatibility. If a vehicle fails to satisfy any one of a mission's requirements, it is excluded from consideration for that mission before the data are input to the algorithm for optimum assignment. Figure 3-5 is a functional diagram for this process which is performed in subroutines CAPABL and AVAIL.

There are four vehicle-to-mission combinations that must be treated: (1) expendable vehicle and no return payload required for mission, (2) reusable vehicle and no return payload required for mission, (3) reusable vehicle and return payload required for mission and (4) expendable vehicle and return payload required for mission. In this analysis, combination (4) is not allowed. Combination (2) is allowed if NTRIP > 0 for that particular mission; otherwise NTRIP = 0 signals that only expendable vehicles may perform that mission. Combinations (2) and (3) are treated like combination (1) using the same general performance check. In this check, if the vehicle can carry the required payload at the required ΔV, then the performance test is passed. Otherwise, the number of trips required by the vehicle to accomplish the mission is computed. If the number of trips required exceeds the maximum number allowed, (input by the user), then that vehicle is eliminated from consideration for that mission.

All vehicles passing the performance test may be further tested by specifying option 3 and the following mission criteria for consideration in the vehicle-to-mission compatibility screen. Actual input form is described in detail in Appendix A.

3-10 3-15 3-10 [illegible]

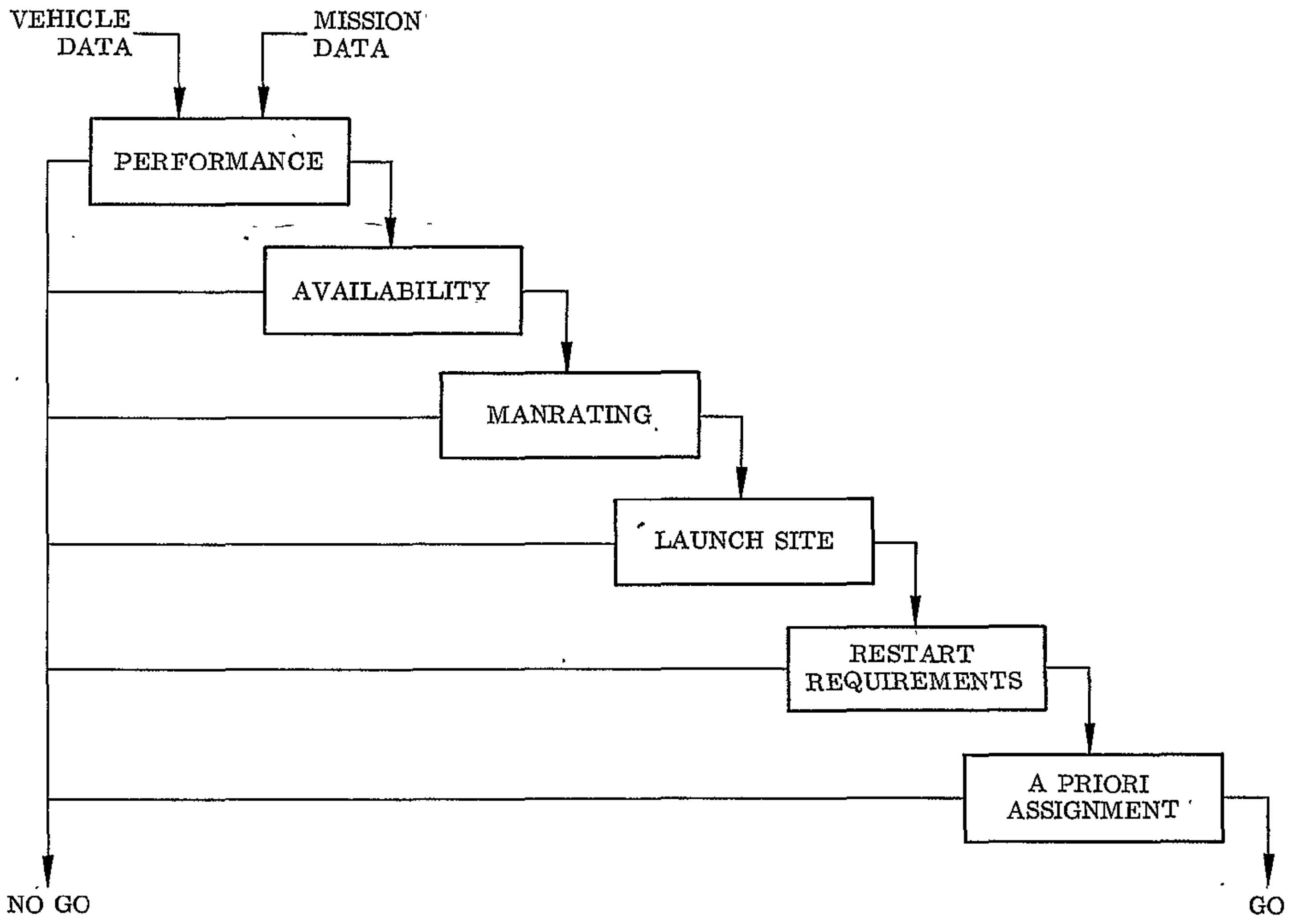


Fig. 3-5 Vehicle-to-Mission Compatibility Screen

(1) <u>Payload Characteristics</u>

   (a) Stabilization requirements (spin-stabilized or not, or no special requirements)
   (b) Man-rating requirement
   (c) A priori vehicle assignment (overrides program determination of optimum vehicle for that mission)
   (d) Weight
   (e) Priority assignment

(2) <u>Launch Characteristics</u>

   (a) Launch site (ETR or WTR)
   (b) Restart requirement (number of restarts required $\leq$ maximum number of restarts possible for vehicle)
   (c) Characteristic velocity requirement
   (d) Launch rate by year

At present the user can specify one of three options of screening for use in determining whether a vehicle can or cannot accomplish a mission.

3.4.1 Option 1

Option 1 first looks to see if an a priori vehicle assignment has been made for any mission. If there is such an assignment, all other vehicles are excluded from consideration for that mission. If no such assignment has been made then Option 1 performs the screening function based on performance criteria and availability. The characteristic velocity (total required mission velocity) and desired payload weight point for each mission is compared to the excess velocity (required total mission ΔV - 25,580 fps) versus payload curve for each vehicle. If the mission requirement point lies below the curve then that vehicle is capable of performing that mission in an initial compatibility matrix. Otherwise a "sizing" test is made as described above. A second check is then made later in subroutine AVAIL to determine if each vehicle is available at the time of launch of those missions which it can perform and a final compatibility matrix is output.

3-12   3-12

3-11 [illegible]

3.4.2 Option 2

Option 2 includes the above screen for any vehicles input directly and a direct performance test for any vehicles formed in the stage-matching screen performed in subroutine MATE and described in subsection 3.7.

3.4.3 Option 3

Option 3 includes the screen described in Option 1, but adds the criteria of stabilization, man-rating, and restart requirements.

For each mission there are three possiblities for the payload stabilization requirement: (1) it must be spin-stabilized, (2) it must be stabilized but not by spin, or (3) there is no requirement. If there is no requirement, then no test of a vehicle's stabilization capability is made. If there is a requirement, however, then a vehicle must meet it or be excluded from consideration for that mission.

The man-rating requirement is handled in a similar manner, except that there are only two possibilities for each mission: it either is man-rated or it is not. If it is not, then no check of a vehicle's capability is made. If it is man-rated, then only vehicles which are man-rated are kept as possibilities for that mission.

The last criterion, the number of restarts required, is simply compared to the maximum number possible for each vehicle and the final decision made on this comparison.

The priority assignment for each mission is a measure, based on a unit scale, of the probability that the mission will be included in future mission models. Thus, the anticipated effects of changing space requirements and the budgetary environment can be imposed on the mission model. The priority factor is multiplied by all recurring costs associated with that mission. Thus, the expected mission recurring cost is output. All other costs are unaffected by the priority assignment.

3.5 REUSABLE LAUNCH VEHICLE CONSIDERATIONS

The multiple parameters associated with reusable launch vehicles such as expected lifetime of each stage, stage turn-around time, mission duration and launch rate, launch pad turn-around time and payload "sizing" requirements may be treated in many ways.

One major use of this assignment program is to determine characteristic values for the above parameters that will make reusable vehicles significantly less expensive than expendables. For this analysis the program currently requires that an initial buy of NU units for each reusable stage be input, along with the unit purchase price, UPP. The total cost of these initial stages is added to the input stage development cost before the algorithm in subroutine CHOOZ determines the optimum assignment. If the cost of the initial stages is included in the input stage development cost, then UPP = unit purchase price is input as zero.

Section 3.4 describes how reusable launch vehicles are screened for mission compatibility, and also "sized" to mission payload requirements if desired. The remaining parameters, that when combined determine the initial buy required for each stage, can be input into the user's own subroutine called REUSE. This subroutine may be as detailed as the user requires but no major changes in the assignment program are necessary for its incorporation. The program has been structured for inclusion of such a subroutine. The MAIN program and subroutine STGNUM contain several comment cards that describe how subroutine REUSE will fit into the current program Thus, the program currently handles initial unit purchases directly, and may easily be expanded by the user so that the optimal number of units to be purchased can be determined for each stage by the program based on the parameters introduced by the user in subroutine REUSE.

3.6 LAUNCH FACILITY AND LAUNCH PAD CONSIDERATIONS

Launch facility and launch pad information may or may not be input as the user desires. If such information is input, then each vehicle is paired with one pad complex at ETR

3-14 3-14 3-13

and one at WTR. Dates of complex availability may also be input; otherwise the program assumes the complex is already available and will be maintained (as required). The assignment program chooses candidates for optimal solution on the basis of one launch pad being available for each vehicle in the candidate set. After each candidate set for optimal assignment is determined, the actual number of launch pads required is computed on the basis of an input maximum number of launches/year/pad possible for each complex. Additional funds required for extra launch pad costs (if any) associated with this candidate assignment are added to the total cost for this assignment. The lower bounds for all competing assignments are compared to this augmented cost. The algorithm proceeds to find new candidate solutions until all competing lower bounds are greater than the actual least total cost computed for one of the candidates already determined. The candidate assignment yielding this least total cost is thus the global optimum solution.

3.7 STAGE MATCHING SCREEN

The capability to combine stages, forming conceptual launch vehicles subject to certain constraints, is available by specifying Option 2 (see subsection 3.4). This option provides identification of combinations of stages that have potential as "acceptable" launch vehicles. Up to 60 launch vehicles comprised of 2, 3, or 4 stages can be generated from stage inputs. Required information to be considered for stages are thrust, weight, diameter, and distance required behind an upper stage by its nozzle structure. Approximate interstage weights are calculated for use in the thrust-to-weight constraints.

A classification is specified for each stage to restrict its usage as follows:

- Type 1 – First stage only (booster)
- Type 2 – Second or third stage
- Type 3 – Second, third, or fourth stages
- Type 4 – Third or fourth stage (uppermost stage on vehicle)

3-15

3-15 3-14

If Option 2 is specified, then the stage information must be input so that Type 1 stages precede Type 2 stages, etc. Stages that are not to be considered in this screen must follow Type 4 stages.

All vehicles generated are subject to the constraint that they can get into low earth orbit as calculated by a simple performance routine. Additional constraints are as follows:

(1) Two-Stage Vehicle
- (a) Thrust-to-weight of first stage less than 3.5 but greater than 1.2
- (b) Diameter of second stage greater than 0.28 but less than 1.2 times diameter of first stage

(2) Three-Stage Vehicle
- (a) Thrust-to-weight of first stage less than 3.0 but greater than 1.2
- (b) Thrust-to-weight of second stage less than 1.25 but greater than 0.37
- (c) Same first- to- second stage diameter constraints as two stage vehicle
- (d) Diameter of third stage greater than 0.28 but less than 1.2 times diameter of second stage

(3) Four-Stage Vehicle
- (a) Thrust-to-weight of first stage less than 3.0 but greater than 1.2
- (b) Thrust-to-weight of second stage less than 1.5 but greater than 0.32
- (c) Thrust-to-weight of third stage less than 1.25 but greater than 0.30
- (d) Same first-to-second and second-to-third stage diameter constraints as three stage vehicle
- (e) Diameter of fourth stage greater than 0.25 but less than 1.2 times diameter of third stage

The above constraints are included because they represent realistic data for launch vehicles of present interest. However, these bounds may be simply varied if later conditions dictate without computational penalty.

3-16 3-16 3-15

As each generated vehicle passes the above screen in MATE it is compared to an input list of vehicles (if any) to see if the same vehicle has already been considered for this mission assignment program. If the generated vehicle is not found on this list, then this new vehicle is added to the list and subroutine PERF is called to determine the capability of this vehicle to perform each mission in the mission model. A flow diagram for subroutine PERF, which was written by the Technical Monitor of this Study, is available in Appendix C of Volume 2. This subroutine is used for vehicles not directly input to the program, and uses the thrust, ISP, empty weight, and fuel weight of all the stages of a vehicle to determine the vehicle's payload for a specified characteristic velocity. It uses an iterative procedure and approximate formulas for the various velocity losses in arriving at a solution. One added feature is that subroutine PERF will use the performance of the lower stages if the curve matching constants of a vehicle using the same lower stages are input.

After all stages have been considered for vehicle formation, or storage has been filled with 60 vehicles (whichever occurs first), the program returns control to MAIN where these vehicles and their capability matrix are further prepared for the algorithm in CHOOZ.

Section 4
BUDGET SMOOTHING AND LAUNCH VEHICLE ASSIGNMENT PROGRAM

The optimum assignment model described in the preceding section was integrated with an existing Budget Smoothing Program described in detail in Ref. 4. The assignment portion of this integrated model includes rate effects on recurring costs but does not include launch pad considerations. The program logic is described in general in this section and is detailed in Appendix G. Complete input requirements are listed in Appendix E along with a glossary of input terms. An interesting sample case is presented in Appendix F which illustrates the form of output and may be used for program check-out. Those portions of the integrated program which duplicate articles found in the preceding section or in Ref. 4 will be omitted from this section.

4.1 LOGIC

The optimum assignment program was integrated with the budget smoothing program through use of a master program which translates from one model to the other. A general logic diagram of this master program and the two main subroutines, ASSIGN and SMOOTH are presented in Figs. 4-1 through 4-3.

The master program (MASTER) calls first the vehicle assignment program (ASSIGN) in order to obtain mission data, cost data and optimum vehicle-to-mission assignment based on this data. MASTER then transforms this data so that it may be used directly by the budget smoothing program (SMOOTH). SMOOTH shifts development dates, launch dates and development duration to achieve a level of spending close to the desired level. The desired levels of spending and constraints on possible program shifts are input to SMOOTH directly. The new development dates and development costs generated by SMOOTH are transformed by MASTER so that ASSIGN can use the data for a revised vehicle to mission assignment. The program iterates between

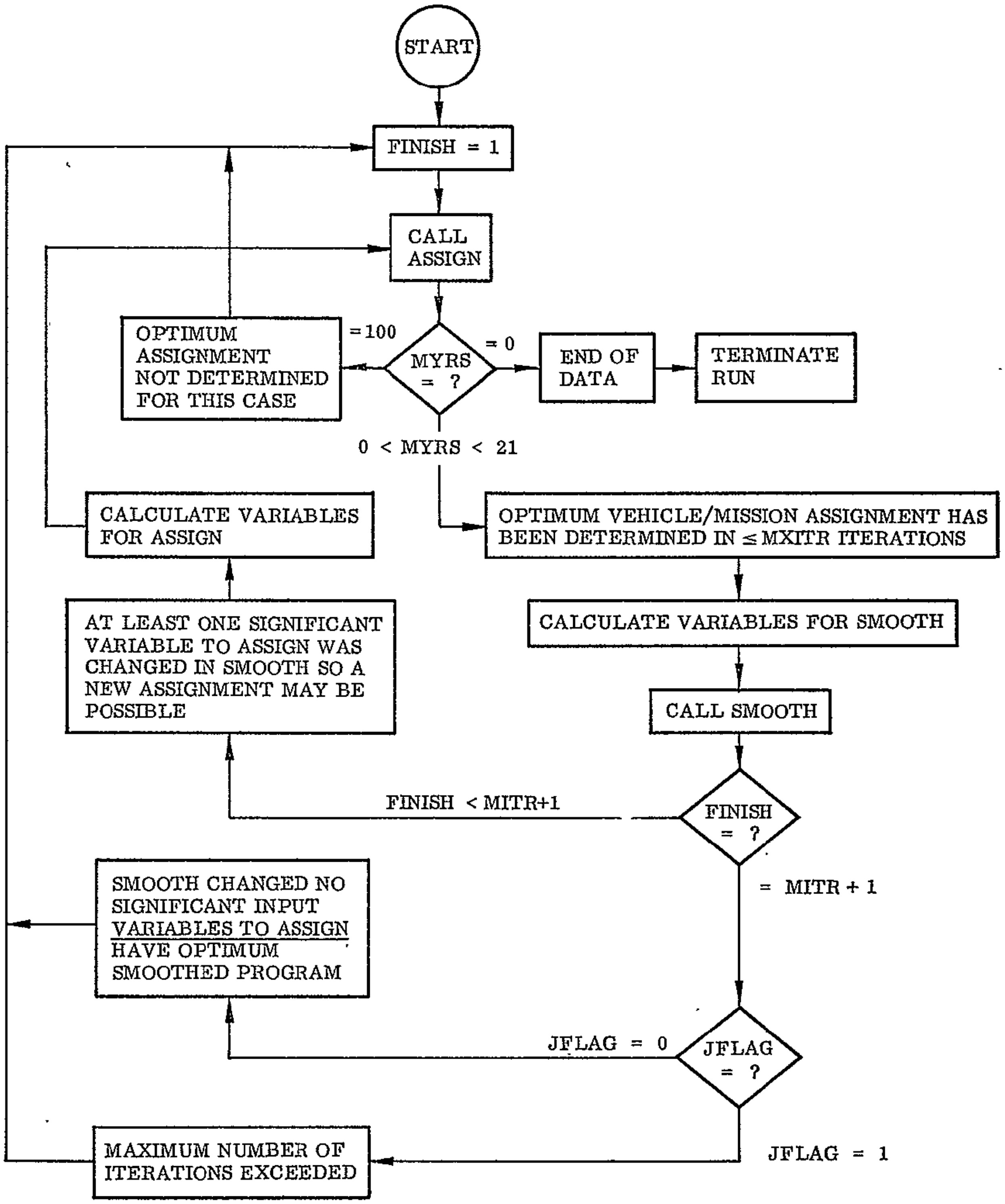

START
FINISH = 1
CALL ASSIGN
MYRS = ?
= 100
OPTIMUM ASSIGNMENT NOT DETERMINED FOR THIS CASE
= 0
END OF DATA
TERMINATE RUN
0 < MYRS < 21
CALCULATE VARIABLES FOR ASSIGN
OPTIMUM VEHICLE/MISSION ASSIGNMENT HAS BEEN DETERMINED IN ≤ MXITR ITERATIONS
AT LEAST ONE SIGNIFICANT VARIABLE TO ASSIGN WAS CHANGED IN SMOOTH SO A NEW ASSIGNMENT MAY BE POSSIBLE
CALCULATE VARIABLES FOR SMOOTH
CALL SMOOTH
FINISH < MITR+1
FINISH = ?
= MITR + 1
SMOOTH CHANGED NO SIGNIFICANT INPUT VARIABLES TO ASSIGN HAVE OPTIMUM SMOOTHED PROGRAM
JFLAG = 0
JFLAG = ?
JFLAG = 1
MAXIMUM NUMBER OF ITERATIONS EXCEEDED

Fig. 4-1 General Flow Diagram for MASTER Program

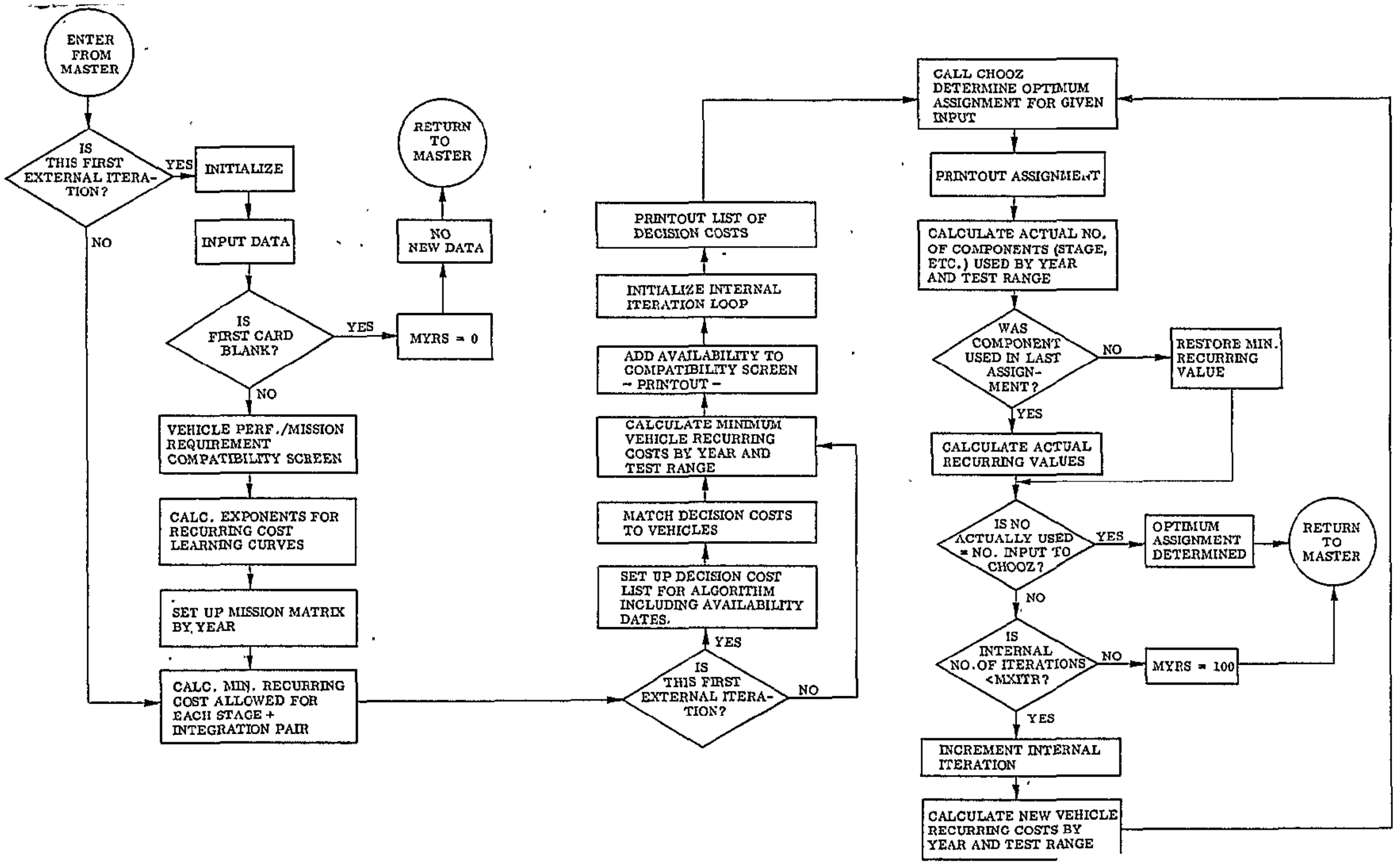
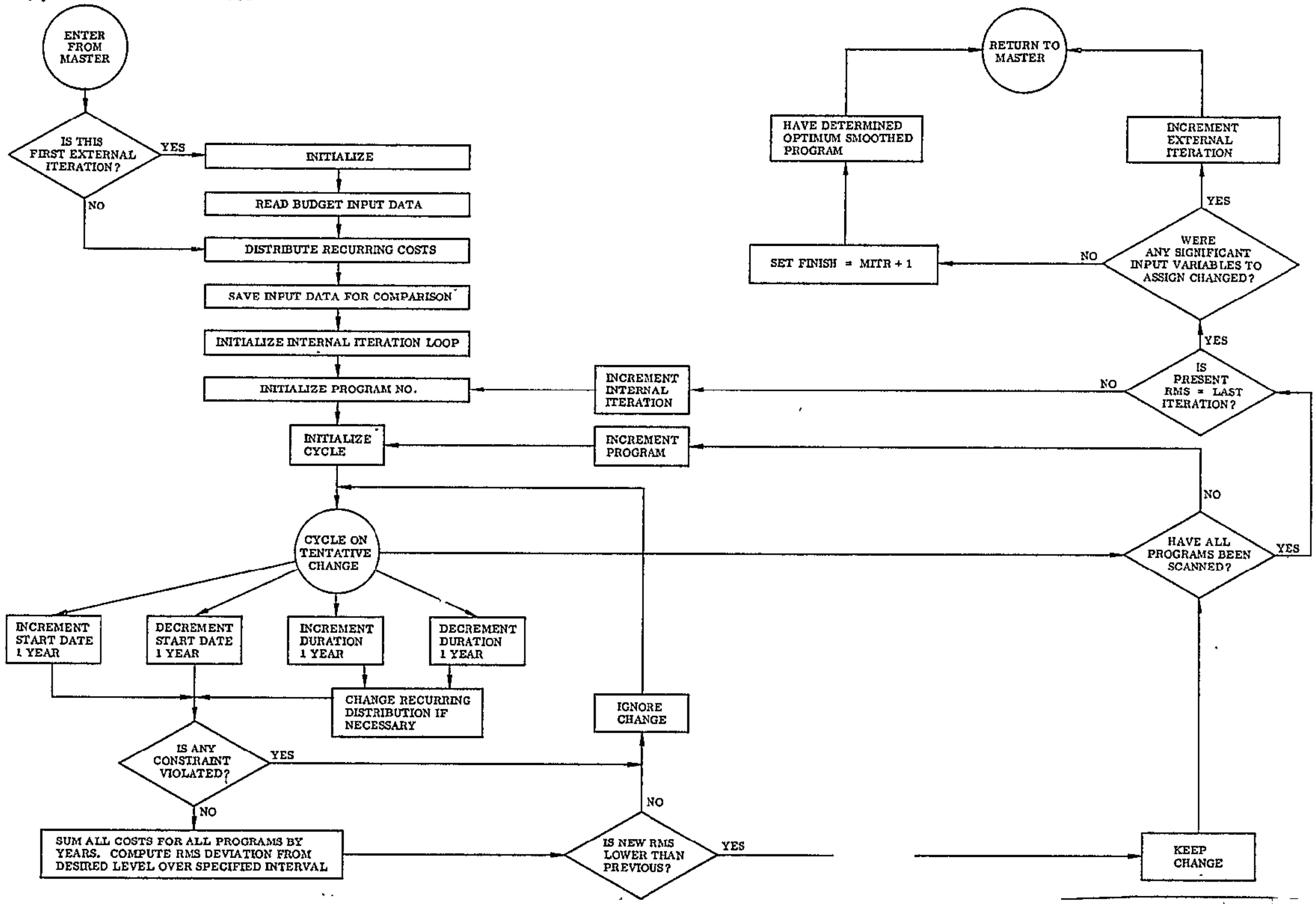

ENTER FROM MASTER
IS THIS FIRST EXTERNAL ITERATION?
YES
NO
INITIALIZE
INPUT DATA
IS FIRST CARD BLANK?
YES
NO
MYRS = 0
NO NEW DATA
RETURN TO MASTER
VEHICLE PERF./MISSION REQUIREMENT COMPATIBILITY SCREEN
CALC. EXPONENTS FOR RECURRING COST LEARNING CURVES
SET UP MISSION MATRIX BY YEAR
CALC. MIN. RECURRING COST ALLOWED FOR EACH STAGE + INTEGRATION PAIR
IS THIS FIRST EXTERNAL ITERATION?
YES
NO
SET UP DECISION COST LIST FOR ALGORITHM INCLUDING AVAILABILITY DATES.
MATCH DECISION COSTS TO VEHICLES
CALCULATE MINIMUM VEHICLE RECURRING COSTS BY YEAR AND TEST RANGE
ADD AVAILABILITY TO COMPATIBILITY SCREEN – PRINTOUT –
INITIALIZE INTERNAL ITERATION LOOP
PRINTOUT LIST OF DECISION COSTS
CALL CHOOZ DETERMINE OPTIMUM ASSIGNMENT FOR GIVEN INPUT
PRINTOUT ASSIGNMENT
CALCULATE ACTUAL NO. OF COMPONENTS (STAGE, ETC.) USED BY YEAR AND TEST RANGE
WAS COMPONENT USED IN LAST ASSIGNMENT?
NO
YES
RESTORE MIN. RECURRING VALUE
CALCULATE ACTUAL RECURRING VALUES
IS NO ACTUALLY USED = NO. INPUT TO CHOOZ?
YES
NO
OPTIMUM ASSIGNMENT DETERMINED
RETURN TO MASTER
IS INTERNAL NO. OF ITERATIONS <MXITR?
NO
YES
MYRS = 100
INCREMENT INTERNAL ITERATION
CALCULATE NEW VEHICLE RECURRING COSTS BY YEAR AND TEST RANGE

Fig. 4-2 General Flow Diagram for ASSIGN Program



ENTER FROM MASTER
IS THIS FIRST EXTERNAL ITERATION?
YES
NO
INITIALIZE
READ BUDGET INPUT DATA
DISTRIBUTE RECURRING COSTS
SAVE INPUT DATA FOR COMPARISON
INITIALIZE INTERNAL ITERATION LOOP
INITIALIZE PROGRAM NO.
INITIALIZE CYCLE
CYCLE ON TENTATIVE CHANGE
INCREMENT START DATE 1 YEAR
DECREMENT START DATE 1 YEAR
INCREMENT DURATION 1 YEAR
DECREMENT DURATION 1 YEAR
CHANGE RECURRING DISTRIBUTION IF NECESSARY
IS ANY CONSTRAINT VIOLATED?
YES
NO
SUM ALL COSTS FOR ALL PROGRAMS BY YEARS. COMPUTE RMS DEVIATION FROM DESIRED LEVEL OVER SPECIFIED INTERVAL
IS NEW RMS LOWER THAN PREVIOUS?
YES
NO
IGNORE CHANGE
KEEP CHANGE
INCREMENT INTERNAL ITERATION
INCREMENT PROGRAM
HAVE ALL PROGRAMS BEEN SCANNED?
YES
NO
IS PRESENT RMS = LAST ITERATION?
YES
NO
WERE ANY SIGNIFICANT INPUT VARIABLES TO ASSIGN CHANGED?
YES
NO
SET FINISH = MITR + 1
HAVE DETERMINED OPTIMUM SMOOTHED PROGRAM
INCREMENT EXTERNAL ITERATION
RETURN TO MASTER

Fig. 4-3 General Flow Diagram for SMOOTH Program

ASSIGN and SMOOTH until no major changes are generated by SMOOTH. Then MASTER either terminates or starts a new case with associated data.

Figure 4-4 illustrates the overall relationship between the 22 subroutines. Subroutines in the assignment portion of the integrated model correspond to subroutines in the assignment model described in the preceding section which have the same name. Common storage in the two versions are different, however, so that subroutines having the same name are stored under different identifying labels. The integrated program consistently uses the first and last letters of the subroutine name for identification. For instance, Subroutine STGNUM is stored under MOX02SM in the integrated program and stored under MOX02SN in the assignment program described in Section 3. Subroutines CLEAR (MOX01CL), INPUT (ALINPT), and PLOT (MOX01UP) are available to all NASA computer users and are described in Ref. 4 and Appendix G. Subroutines PACK (MOX01PK) and AFRMT (MOX02AT) were written in 360 Assembler Language by the Technical Monitor of this study. Listings for each are included in Appendix H and a description of both subroutines appear in Appendix G. The remaining subroutines have flow charts in detail in Appendix G and Fortran listings in Appendix H. The first comment card in each subroutine listing states the primary purpose of that subroutine. Other comment cards describing the purpose of each section and defining any pertinent variable whose name is not mnemonic are distributed liberally throughout the listing so that new users may familiarize themselves with the logical function of each subsection within the program.

Dimension restrictions are detailed in Appendix E for input variables and for internal variables indirectly associated with the input. Other internal dimension restrictions may be found in the first part of the program listing for MASTER in Appendix H. Equivalence relations and data statements are also listed in MASTER.

4-5

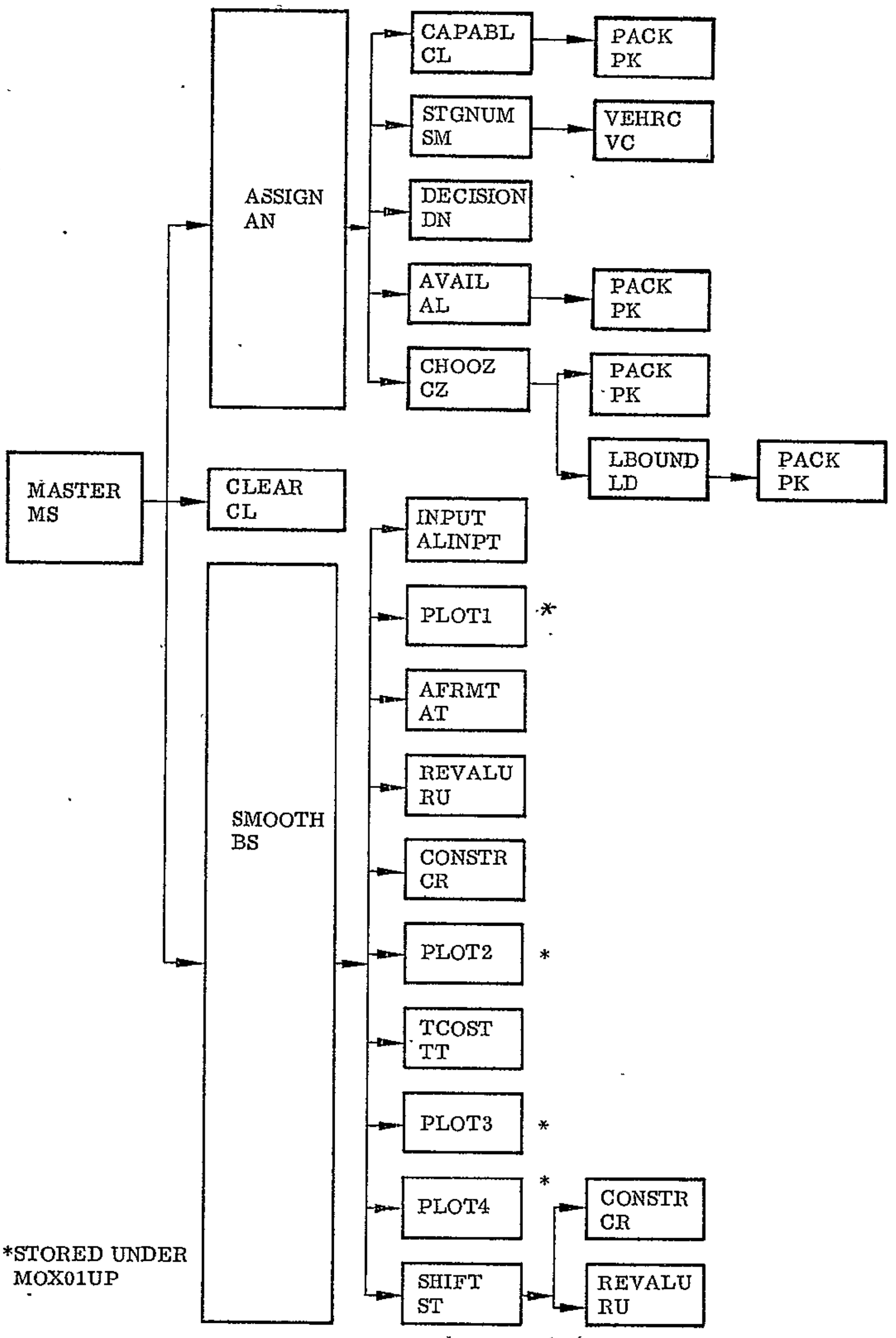


Fig. 4-4 Program Subroutine Relationships

4.2 INPUT REQUIREMENTS

Detailed input requirements are available in Appendix E and the general input philosophy will be discussed in this subsection.

In the integration with the Smoothing Program, the Assignment Program described in Section 3 was essentially unchanged. Only new input variables were added to the original input section. These new variables are not used explicitly in the algorithm which determines the least cost assignment; they are, however, carried back to the Master Program where they are modified, if necessary, before entering the Budget Smoothing subroutine. This procedure provides better organization of input data and allows data changes to be made easily.

The Budget Smoothing Program described in Ref. 4 was modified to reflect the changed input philosophy. Essentially only the budget levels, smoothing intervals and program constraints are now input directly to the SMOOTH subroutine.

Internal variables have the same definition in the two versions of the Budget Smoothing Program (i.e., original version in Ref. 4 and modified version in Subroutine SMOOTH). Thus, reference from one to the other is facilitiated.

4.2.1 Program Definitions

The original Budget Smoothing Model is extremely flexible in the type of input it can receive. In order to retain as much generality as possible and still interface the two programs the following approach was used. A program, whose elements are to be shifted by SMOOTH, is defined to be either:

(1) Missions involving at least one launch date and no more than 10 consecutive launch years per specific mission. Payload costs (including recurring, development and sustaining types) and corresponding dates of expenditure are input along with the mission requirements and launch rate schedule.

(2) Development and sustaining programs associated with feasible launch vehicles. Recurring costs and their associated rate parameters are input at the same time as expenditure distributions for each such development and sustaining program. These development programs may be related directly to a stage, integration, or shared cost group. In contrast, the physical characteristics of any vehicle and the distribution of the recurring cost associated with that vehicle is input along with other strictly vehicle-related information.

(3) Any miscellaneous program having no direct effect on the optimum vehicle assignment and no launch schedule associated with it, but having a definite development and/or sustaining and/or fixed cost associated with it. (e.g., a future development program whose costs and dates of expenditure are known approximately but which can be tied to no definite single mission or vehicle component.)

Program types (1) and (3) have all associated costs input to the SMOOTH subroutine, while only those development and sustaining programs in (2) actually selected in the vehicle assignment algorithm are applicable to the SMOOTH subroutine. Only type (2) costs are used to determine the optimum vehicle assignment in the combined program.

4.2.2 Program Elements

Typical program elements are illustrated in Fig. 4-5 for program types (1) and (2). Existing stages are sustained from the input reference year until the last year they are actually used as determined by the algorithm in ASSIGN. Sustaining costs for newly developed stages begin after the development is two-thirds complete. Stages are considered operational during their last year of development. Vehicle recurring costs are spread by the standard distribution (first year = 0.05 total recurring cost, second year = 0.20 total recurring cost, third year = 0.50 total recurring cost, fourth year = launch year = 0.25 total recurring cost) unless overridden by input

4-8 4-8 47

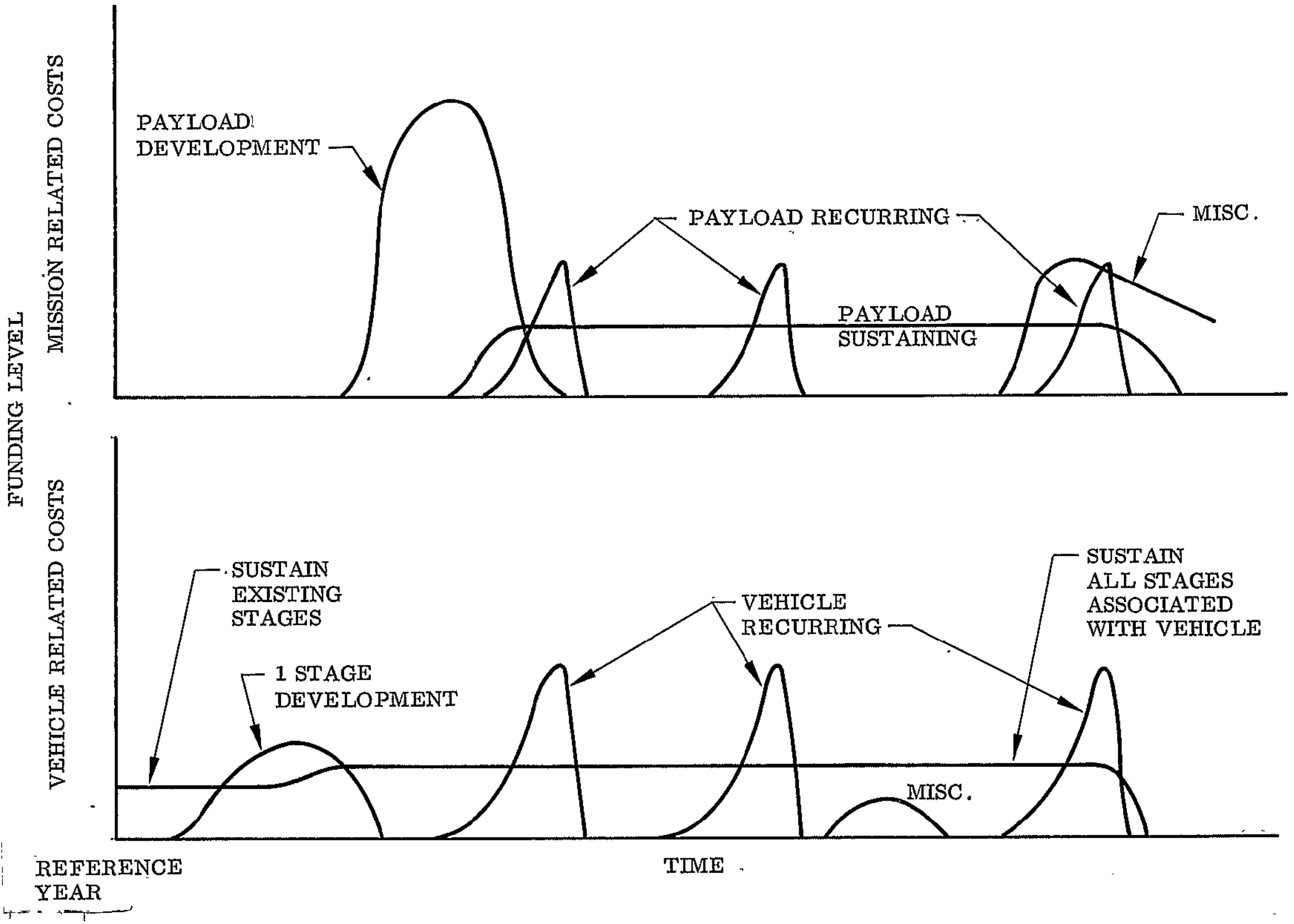


Fig. 4-5 Typical Program Elements

into variable ALPI (see Appendix E). Development costs associated with a vehicle (whether they are stage, family or integration related) are automatically distributed by the symmetric Beta distribution over the development duration input with each such cost.

If another distribution is desired, that cost (whether it be development, run-out or any other miscellaneous cost) may be input under the variable name RXD with its associated start date, NRFX, and duration in years, NSFX. Since these miscellaneous costs are input as fixed dollar amounts for each specified year, any distribution desired is allowed as long as its duration is less than 13 years.

Payload and other mission related costs are input in much the same way as vehicle related costs. The two major differences are that

(1) Payload recurring costs are distributed by an input distribution to RDIST only. There is no automatic standard distribution.
(2) The payload may be sustained for NSYR years after the last launch. This provision allows for data acquisition and compilation.

It should be noted that miscellaneous costs associated with a mission or launch vehicle component are constrained in SMOOTH by any input constraints on the associated program. For example, if the development start date of a mission payload is constrained, then the start date of any miscellaneous cost associated with that mission is similarly constrained. In contrast, program type 3 costs (see subsection 4.2.1) which are miscellaneous in nature are only constrained by direct inputs to that effect.

4.2.3 Constraints

Constraints are input directly to SMOOTH involving program types (1) and (3). They are keyed according to the following table where:

KODE	= the type of constraint by KEY number (see Table 4-1)
NPROG	= N = the constrained program reference number
KPROG	= K = the constraining program reference number
CS	= associated real number constant

4-11 4-11 4-8

Table 4-1

KEY TO PROGRAM CONSTRAINTS(a)

KODE	Type of Constraint
1	$START_N > END_K + CS(1)$
2	$END_N + CS < START_K (1)$
3	$START_N = CS$
4	$END_N = CS$
5	DEV. $DURATION_N = CS$ (FIXED DURATION)
6	LAUNCH $DATE_N + CS \leq$ LAUNCH $DATE_K$
7	LAUNCH $DATE_N \leq CS$
8	NO CHANGES ALLOWED
9	$START_N \geq CS$
10	LAUNCH $DATE_N \geq CS$
11	$END_N + CS <$ LAUNCH $DATE_K$

(a) START and END refer to Development

Examples of the flexibility of this approach are provided in Ref. 4. Input program data must satisfy the input constraints to ensure a correct output from SMOOTH. Any violations in input data are printed out before "smoothing" begins so that the user is aware of the condition. The program will continue even if violations occur since in many cases the violations are corrected by the "shifting" process.

Program type (2) data is automatically constrained in the MASTER routine. KODE 11 is used to ensure that all development programs selected by ASSIGN in the optimum solution end before the component being developed is to be launched. Thus, SMOOTH is automatically constrained so that the optimum vehicle assignment input to SMOOTH is still a feasible candidate assignment after SMOOTH is complete. Whether this assignment is still optimum depends on which variables have been "shifted" by SMOOTH. If key variables have been changed ASSIGN is called again to determine a new (or possibly the same) optimum assignment.

4.3 OUTPUT

A sample output is presented in Appendix F and a general description follows. First, all the input data is output for reference, including data computed by the program which will be input to the ASSIGN algorithm. Then the optimum assignment is output listing each mission and the assigned optimum vehicle, along with total mission model cost.

Input to SMOOTH is output automatically as it appears on the data card. "Average" recurring cost data for each vehicle in the optimum assignment is computed in VEHRC. It is determined by totaling the actual recurring cost for each vehicle over the entire mission duration and then dividing by the total number of vehicles used throughout the mission model. Finally the input cost data is output by program and type and also by year. A plot showing actual spending by year and desired spending level by year follows. The program then smooths this input data and outputs the final result in the same form as it did the input data. Launch vehicle requirements by year are output using the smoothed data. At this point the program either terminates because an optimum smoothed assignment has been found or else it returns to ASSIGN and outputs the new data which will be used in the algorithm. The output cycle then continues as explained above until an optimum solution has been found.

Section 5
CONCLUDING REMARKS

5.1 GENERAL DISCUSSION

In performing advanced planning for space missions for an extended future period, numerous factors affect the selection of launch vehicles and their assignment to missions for a least-cost total program. These many factors must be evaluated by the planner in accomplishing the maximum space program under continuing economic constraints. In addition, the planner must consider year-by-year funding limits and the priority of missions consistent with national goals, and be able to rapidly develop and assess alternative programs.

A future space program will normally include a mix of existing stages or vehicles, growth versions of these, and new starts. Assessment of the optimal application of this mix to future programs requires evaluation of associated vehicle physical, performance and cost factors. Similar data must be considered as they relate to production, operation and maintenance, and program elements such as launch sites and pads, manufacturing lines, and sustaining engineering. Because of continuing budgetary restrictions and complexity resulting from interrelationship between the many cost factors, the economic aspects are very important. The extensively modified branch-and-bound algorithm, which is the core algorithm of the developed programs (the optimal vehicle assignment model and the integrated budget smoothing and vehicle assignment model, emphasizes this economic analysis.

5-1

5.2 SUMMARY OF RESULTS

The following selected comments are based on the preceding summary discussion and detailed description in the report:

- To achieve optimal least-cost assignment of launch vehicles to future space missions over an extended future period, many factors, including physical, performance, and cost parameters, must be considered.
- The interrelationships between these many factors results in a large combinatorial problem. Because of its dimensions, the problem has not been tractable to direct methods — manual or computer based. The analytical approach and implementing computer programs developed in this study, therefore, use accelerated search techniques in providing the optimal vehicle-to-mission assignment.
- Budgetary constraints are of continuing importance to future space programs. While satisfying physical and performance requirements in the vehicle-to-mission assignment, therefore, emphasis has been given to economic considerations in the solution methodology. In this methodology all cost elements are explicilty handled in the three basic categories (nonrecurring, recurring, and sustaining), and as a result a global, minimum-cost vehicle assignment is assured.
- Techniques have been emphasized that decrease computation time. These include the use of penalty functions in making node decisions and lower bound estimates, and bypassing unnecessary subroutines during core algorithm operation.
- "Packing" and "overlaying" techniques have been used to permit operation on computers with storage limitations.
- Each vehicle candidate analyzed can consists of four stages. Using alternate configurations of the same stage, the program can compare alternate designs.
- Both the vehicle assignment and the integrated budget smoothing and vehicle assignment programs are structured to provide independence between subroutines. This characteristic provides the user with flexibility in applying the programs to his particular problem.

- "Shared or "family" costs arise from derivative versions of a vehicle—e.g., the "Atlas family," "Titan family," and "Agena family." In these cases certain costs are intradependent and are handled appropriately by the models.
- The integrated vehicle assignment and budget smoothing program provides a powerful tool for the advanced planner. With data input for missions and stages (or vehicles) under consideration, the model can assess various alternatives and provide data for planning decision. Variations of interest include different budget ceilings, mission priorities, alternative mission models, constrained launch windows, and other factors. Interaction between the vehicle assignment model and the budget smoothing model ensures that a least-cost program is provided for smoothing in each case.
- On a single run an optimal vehicle-to-mission assignment, the total program cost and an array of data relating to this least-cost program is output. By iterative runs, sensitivity to variations in input parameters can be determined. While run time is normally short, it is desirable to assess the effect of input variability on a single-run basis so that a parametric family of outputs can be provided by a single run. It is recommended that the model be modified to provide this capability.

Section 6
REFERENCES

1. J. D. C. Little et al., "An Algorithm for the Traveling Salesman Problem," Opns. Res. 11, 972–979 (1963)

2. E. L. Lawler, and D. E. Wood, "Branch-and-Bound Methods: A Survey," Opns. Res. 14, 699–719 (1966)

3. P. Bertier and B. Roy, A Solution Procedure for a Class of Problems Having Combinatorial Character, (Translated by William S. Jewell), ORC 67-34, Operations Research Center, U. of Calif., Berkeley (1967)

4. R. E. Slye, "Budget Smoothing Model for Program Planning," (to be released as a NASA TN), NASA Mission Analysis Division, Moffett Field, California

6-1

[illegible]